

关于CSS方面的分享

汇集四届CSS大会的内容 (2015/1 - 2018/3)

烈风袭

2018/04

“如果Slides太多，一次报告讲不完，那就挑些思路讲讲。”

— 鲁迅

BIG MAP



❶ 组件

组件

事实与共性

在**某个框架**上开发组件库是现在的主流，比如React和Vue。

组件需要复用，理解**结构与表现分离**的思想。

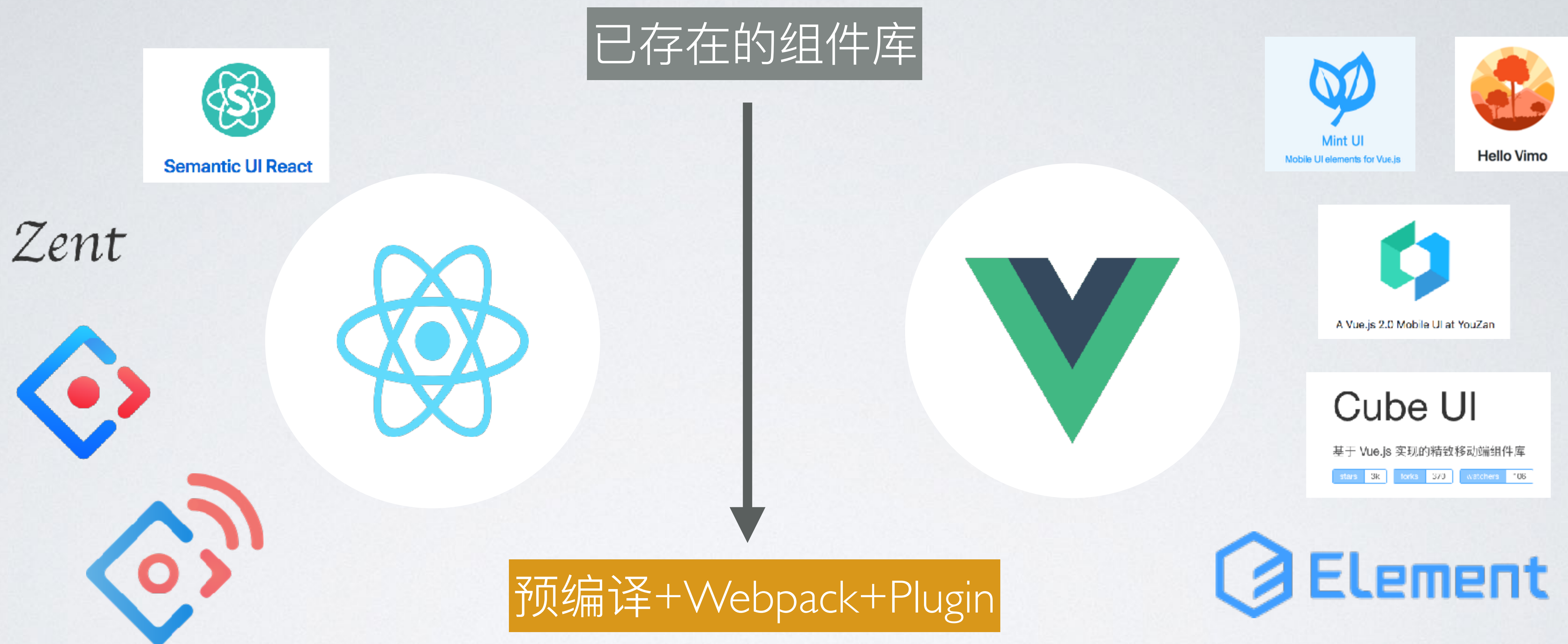
组件库的样式系统应该满足**继承模型**。

组件库样式编写需要一个**书写规范**。

组件库必须**暴露丰富的样式接口**。

- 第四届 Zell Liew [Responsive components](#)
- 第四届 顾轶灵 [可复用组件的CSS接口设计](#)
- 第三届 廖洧杰 [Sass & CSS Design Pattern](#)

组件



- 第四届 Zell Liew Responsive components
- 第四届 顾轶灵 可复用组件的CSS接口设计
- 第三届 廖洧杰 Sass & CSS Design Pattern

组件

结构与表现分离

babel-plugin-vimo

```
import { Button } from 'vimo';
```

转化为:

```
"use strict";  
require("vimo/components/button/button.scss");  
require("vimo-theme-md/components/button.md.scss");  
require("vimo-theme-ios/components/button.ios.scss");  
  
var _button2 = _interopRequireDefault(require("vimo/components/button"));  
function _interopRequireDefault(obj) {  
    return obj && obj.__esModule ? obj : { default: obj };  
}
```



同质问题能用插件解决的
别手动编码!

- babel-plugin-vimo
- babel-plugin-import
- babel-plugin-component

组件

事实与共性

在**某个框架**上开发组件库是现在的主流，比如React和Vue。

组件需要复用，理解**结构与表现分离**的思想。

组件库的样式系统应该满足**金字塔模型**。

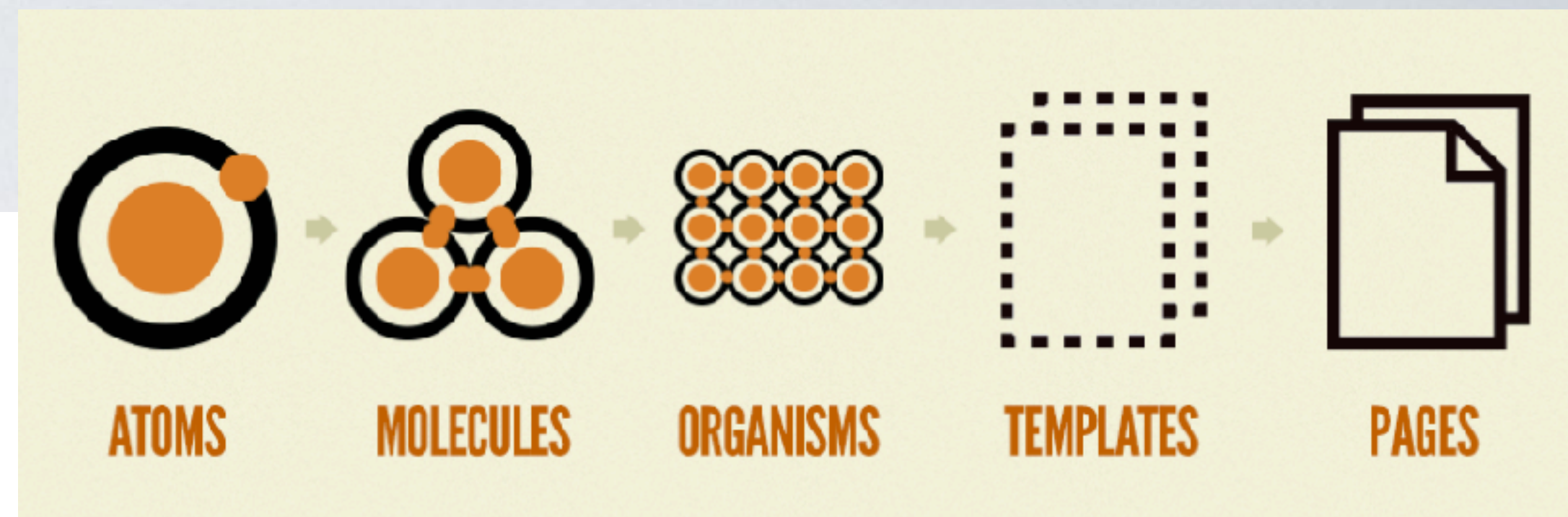
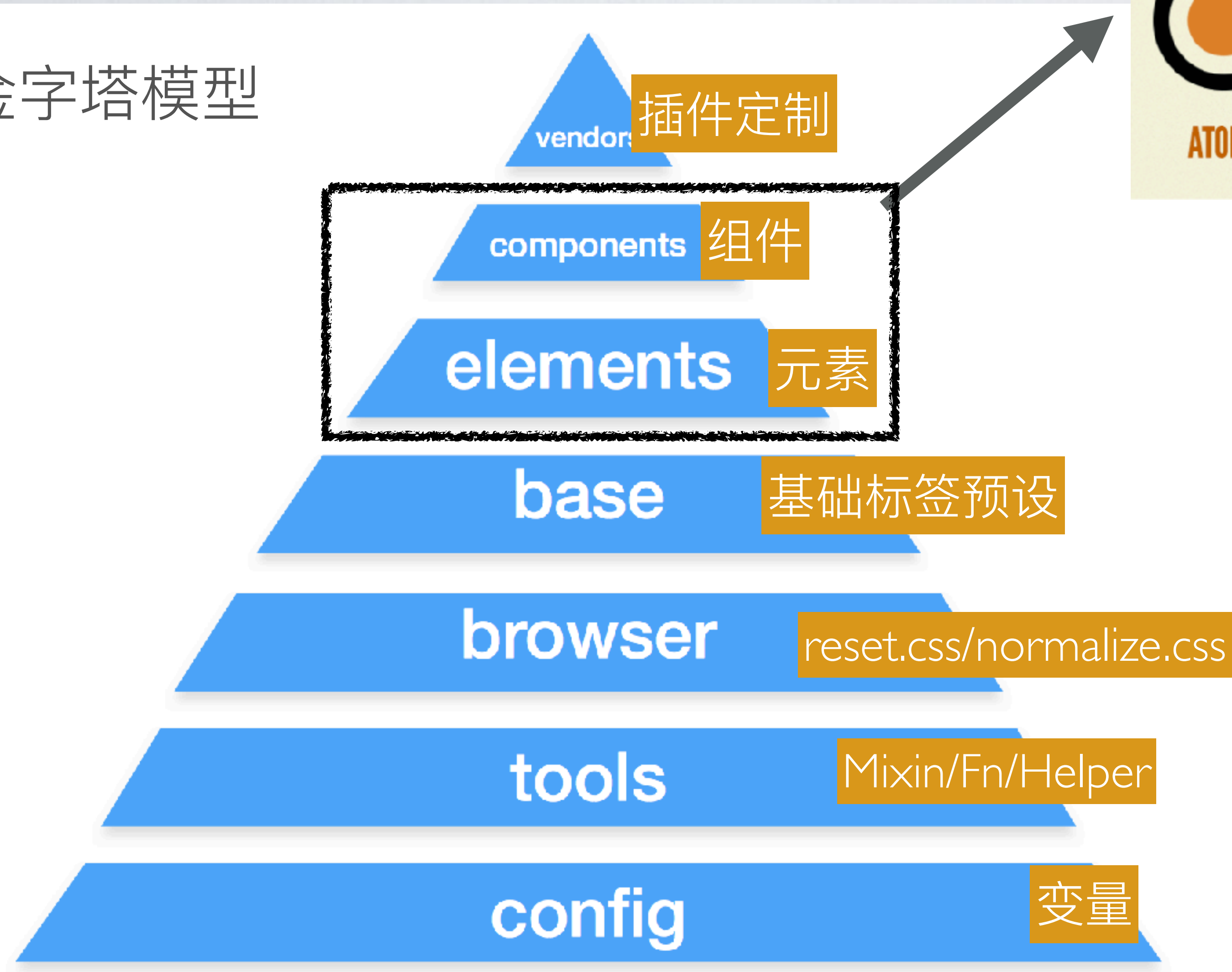
组件库样式编写需要一个**书写规范+预处理器**。

组件库必须**暴露丰富的样式接口**。

- 第四届 Zell Liew [Responsive components](#)
- 第四届 顾轶灵 [可复用组件的CSS接口设计](#)
- 第三届 廖洧杰 [Sass & CSS Design Pattern](#)

组件

金字塔模型



书写规范: **BEM(30%+)**

预处理器: **SCSS(63.3%)/
Less/PostCSS/Stylus**

- 第四届 顾轶灵 可复用组件的CSS接口设计
- 第三届 廖洧杰 Sass & CSS Design Pattern
- BEM模式
- Atomic Design

组件

事实与共性

在**某个框架**上开发组件库是现在的主流，比如React和Vue。

组件需要复用，理解**结构与表现分离**的思想。

组件库的**样式系统**应该满足**金字塔模型**。

组件库样式编写需要一个**书写规范+预处理器**。

组件库必须**暴露丰富的样式接口**。

- 第四届 Zell Liew [Responsive components](#)
- 第四届 顾轶灵 [可复用组件的CSS接口设计](#)
- 第三届 廖洧杰 [Sass & CSS Design Pattern](#)

组件

用别人的组件没自由，我要改

Props属性

使用预处理器中的变量

CSS优先级覆盖

*CSS-IN-JS的API (JSS)

- 第四届 Zell Liew [Responsive components](#)
- 第四届 顾轶灵 [可复用组件的CSS接口设计](#)
- 第三届 廖洧杰 [Sass & CSS Design Pattern](#)
- [关于组件库主题定制](#)
- [CSS优先级计算及应用](#)

```
<el-button type="primary">主要按钮</el-button>  
<el-button type="success">成功按钮</el-button>
```

```
/* 改变主题色变量 */  
$--color-primary: teal;  
  
/* 改变 icon 字体路径变量，必需 */  
$--font-path: '~element-ui/lib/theme-chalk/fonts';
```

组件

事实与共性

在**某个框架**上开发组件库是现在的主流，比如React和Vue。

组件需要复用，理解**结构与表现分离**的思想。

组件库的**样式系统**应该满足**金字塔模型**。

组件库样式编写需要一个**书写规范+预处理器**。

组件库必须**暴露丰富的样式接口**。

- 第四届 Zell Liew [Responsive components](#)
- 第四届 顾轶灵 [可复用组件的CSS接口设计](#)
- 第三届 廖洧杰 [Sass & CSS Design Pattern](#)

2 响应式

2 响应式

一般to C会考虑到响应式

移动端rem布局方案
移动端vw布局方案

等比缩放 \neq 响应式

vmax、vmin
vw、wh
ch
rem、em
media query

- 第四届 Zell Liew [Responsive components](#)
- 第四届 一丝 [面向设计的CSS](#)
- [CSS参考手册](#)

2 响应式

VW的支持度

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android *	Chrome Android
			49						
			63						
		58	64			10.3		4.4	
² 11	16	59	65	11	50	11.2	all	62	64
	17	60	66	11.1	51	11.3			
		61	67	TP					
			68						

移动端可以试试
Android 4.4+(2013)
IOS 8+(2014)

是时候试试 vw 了

```
.banner {  
  display: flex;  
  .px2vw(height, 50);  
  .px2vw(margin-bottom, 10);  
}
```

如果能用插件解决
就完美了

postcss-px2vw

- 第四届 Zell Liew [Responsive components](#)
- 第四届 一丝 [面向设计的CSS](#)
- [CSS参考手册](#)

3 编译器

CSS的问题



1. 属性之间的生效关系

2. 副作用

全局
基础规则变更
命名冲突
子树匹配

3. 缺少抽象能力（架构）

看书/实践

BEM

The Stylus logo, featuring the word 'Stylus' in a bold, blue, sans-serif font.

The Sass logo, featuring the word 'Sass' in a purple, cursive script font.



The 'less' logo, featuring the word 'less' in a white, sans-serif font inside a dark blue rounded rectangle.

The MCSS logo, featuring the text 'MCSS' in a bold, blue, sans-serif font on a white background.

- CSS的主要缺陷是什么?
- PHILIP WALTON Side Effects in CSS 2015/3/3
- GetBEM

CSS预处理器



变量 -> 定制开发

嵌套 -> 选择器精确匹配

函数/Mixin -> 复用部分封装

继承 -> 复用

import -> 外部引入

逻辑循环 -> 简化书写

3

编译器

精通C++不会Java也没关系

学会一个剩下的都是语法问题，不难

编译器

预编译器、CSS前处理器、CSS后处理器、PostCSS, 晕了🌀

郑海波 (2015)

- 后处理器应该作用在是在运行时之后的: 如fixed-sticky, prefixfree
- **postcss完全符合css预处理器的定义**

一丝 (2018)

PostCSS 既不是预处理器也不是后处理器



PostCSS
@PostCSS



正在关注

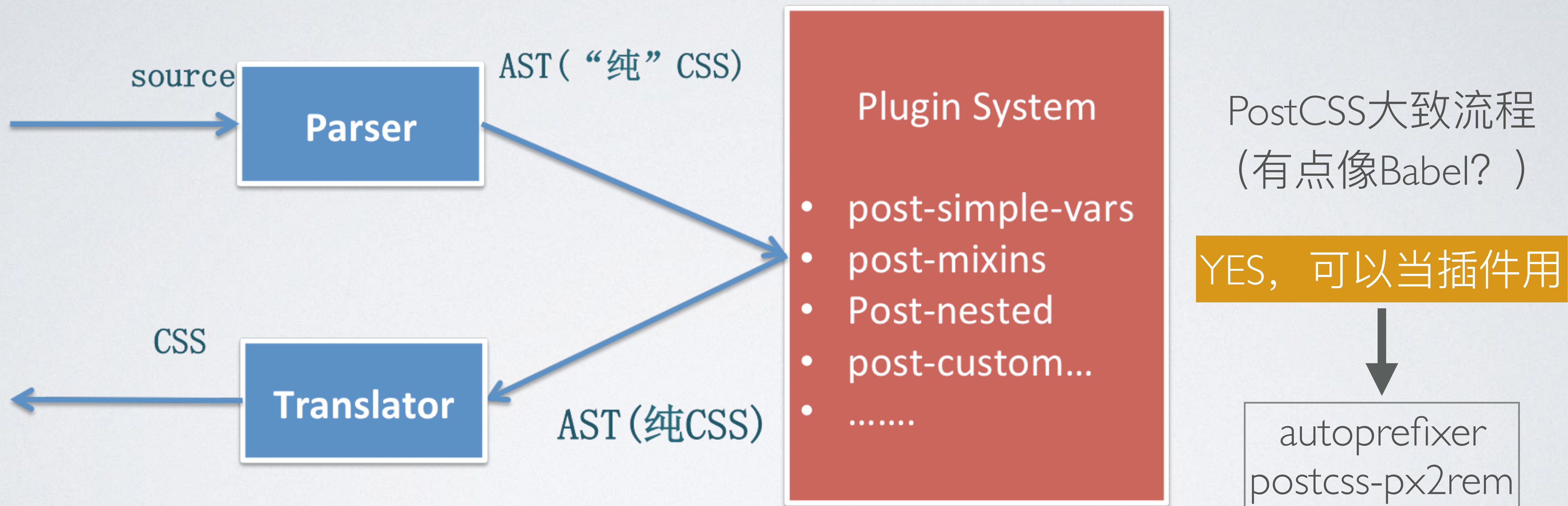
It is time admit my mistakes. "Postprocessor" term was bad. PostCSS team stoped to use it.

[github.com/postcss/postcs...](https://github.com/postcss/postcss)

我不较真
因为我不关心
Hah..

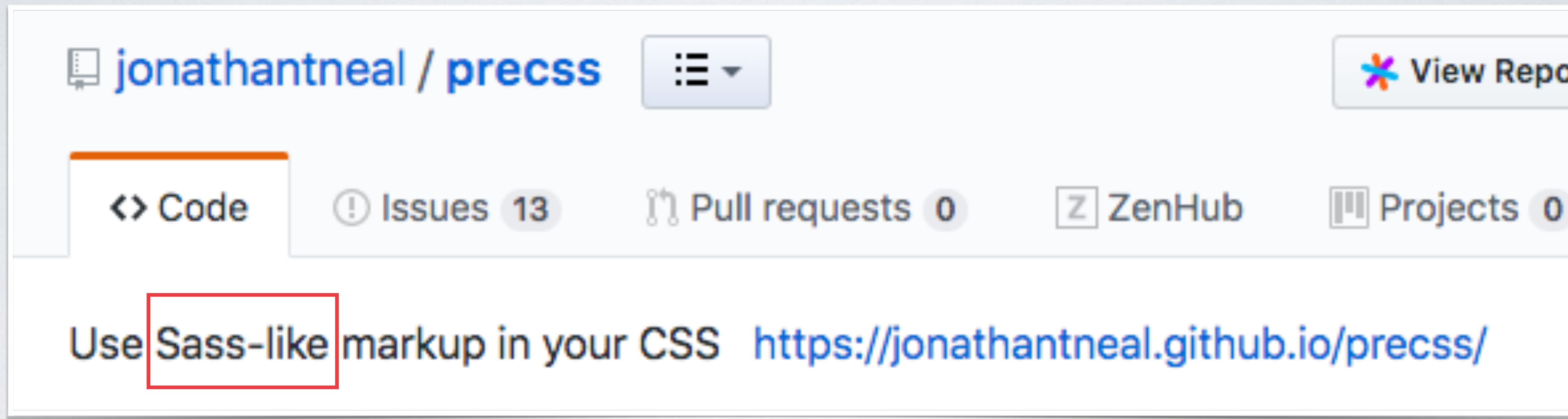
- 第四届 一丝 面向设计的CSS
- 第二届 郑海波 CSS预处理器老调重弹

聊下PostCSS



- 第二届 郑海波 CSS预处理器老调重弹
- PostCSS

也可以当预编译器用



- [第四届 一丝 面向设计的CSS](#)
- [第二届 郑海波 CSS预处理器老调重弹](#)
- [PostCSS](#)
- [PreCSS](#)



不必大费周章，用**Sass/Scss**吧

支持CSS下一代标准 (阉割的)

Use Future CSS, Today

- `autoprefixer` adds vendor prefixes, using data from Can I Use.
- `postcss-preset-env` allows you to use future CSS features today.
- `postcss-image-set-polyfill` emulates `image-set` function logic for all browsers

- 第四届 一丝 面向设计的CSS
- 第二届 郑海波 CSS预处理器老调重弹
- PostCSS

PostCSS值得尝试

POSTCSS的意义

- 更接近CSS
- 可靠&标准化的配套,降低进入成本
 - 入口: AST解析 以及帮助函数
 - 出口: SourceMap & 翻译器
- 100%的插件化, 且极具规模
- 速度更快: 3x-30x
- 提高对新规范的关注度

但是

规范可能变动

预处理能解决的一般不会发布为新规范

Polyfill实现不完全

总结

SCSS和LESS挑一个深入学习，PostCSS必须关注

4 动画

Web开发没有动画就没意思了

动效开发模式

动画性能

动画属性
Transform/
Animate/Matrix

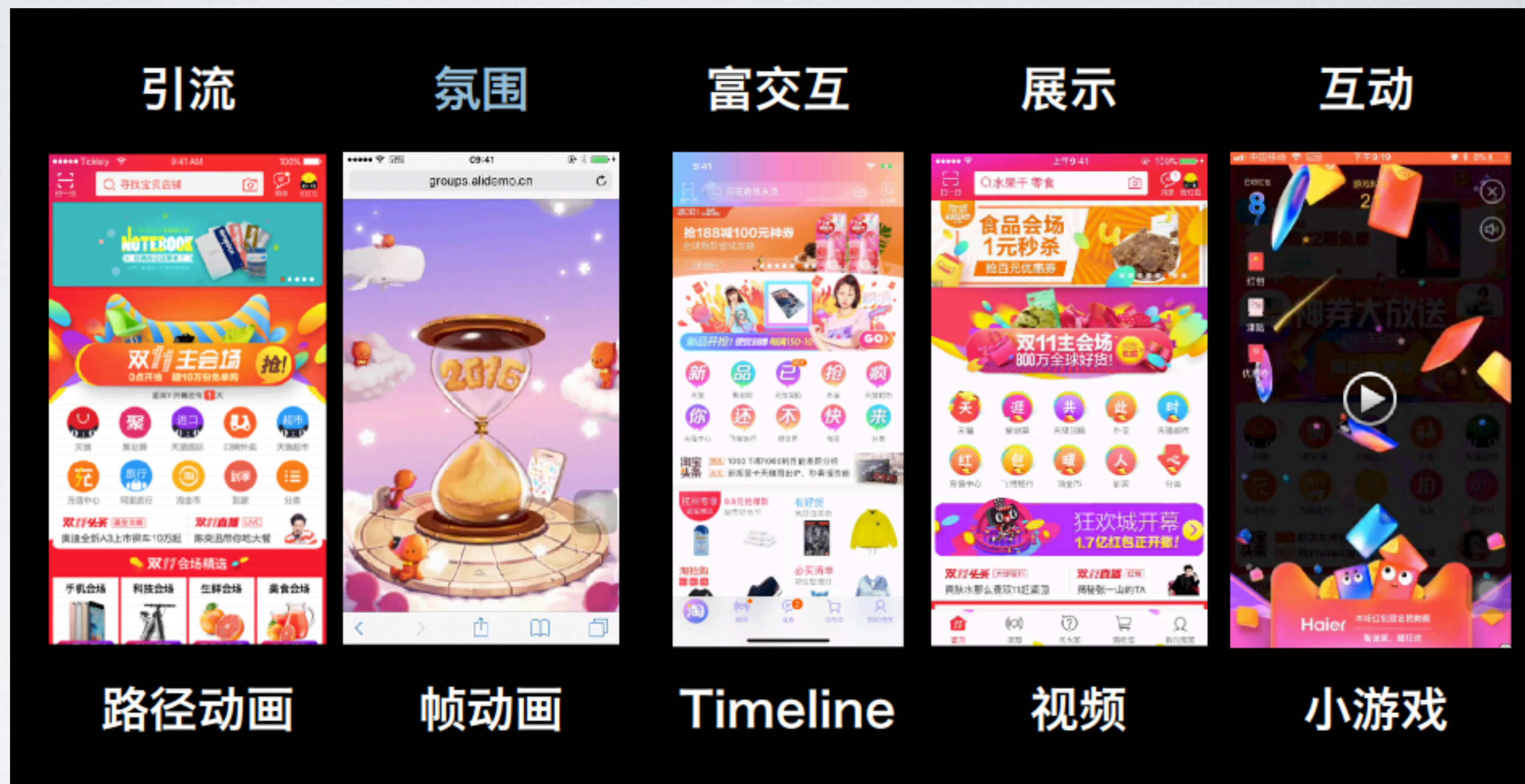
SVG动画

动画属性
Canvas/WebGL

- 第四届 大漠 探索动效开发模式
- 第三届 陈剑鑫 从矩阵走入 WebGL 世界
- 第三届 方潇仪 SVG动画实践

- 第二届 黄薇 高性能CSS动画
- 第一届 吴小倩 谈谈CSS性能
- 第一届 尤雨溪 CSS 与界面动效

动效分类



展示型动画可以不需要开发介入（需要给力的团队）

动效开发模式

大会上讲的PPT在多个地方讲过，没新意

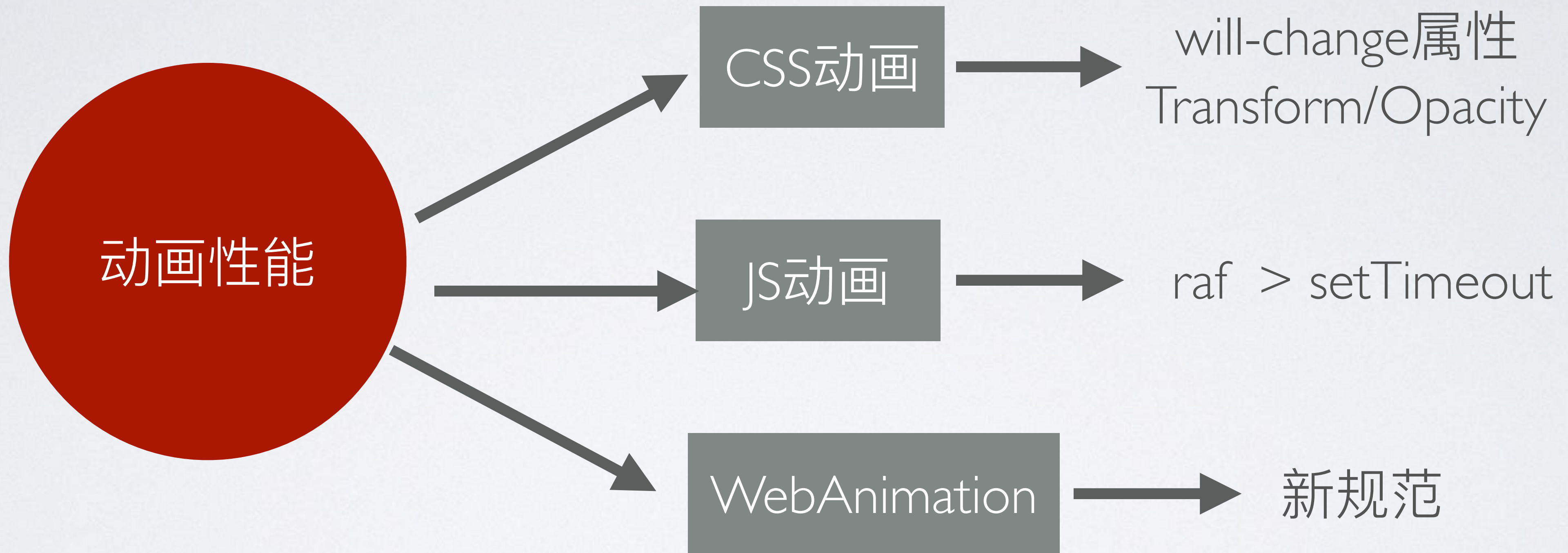


开发模式变迁

```
graph LR; A[动画设计] --> B[视觉导出 (JSON)]; B --> C[发布上线]
```

The diagram illustrates a three-step process for animation development. It starts with '动画设计' (Animation Design) in a green box, followed by '视觉导出 (JSON)' (Visual Export (JSON)) in a yellow box, and finally '发布上线' (Release and Go Live) in a blue box.

CSS/JS动画—老生常谈第一季



- 第二届 黄薇 高性能CSS动画
- 第一届 吴小倩 谈谈CSS性能
- 第一届 尤雨溪 CSS 与界面动效

CSS/JS动画—老生常谈第二季

	CSS	JS
优点	不占用JS主线程	可控
缺点	中间状态不可控 无法新增动画	性能、卡顿
注意点	使用动画属性 创造Layer will-change	使用raf 每帧控制在10ms内
共同注意点	避免Layout, 减少Repaint	

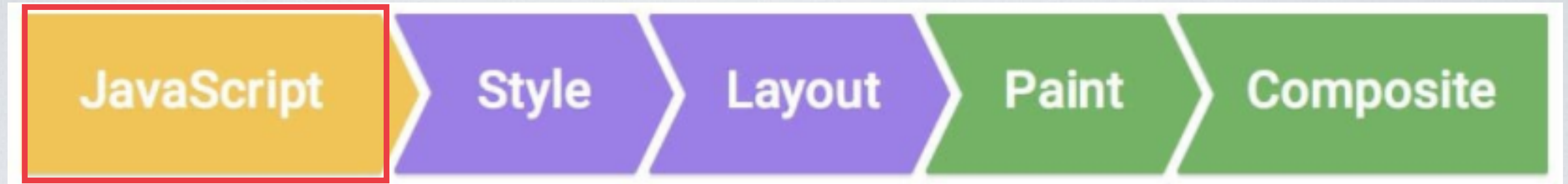
根据使用场景
选解决方案

- 第二届 黄薇 [高性能CSS动画](#)
- 第一届 吴小倩 [谈谈CSS性能](#)
- 第一届 尤雨溪 [CSS 与界面动效](#)

动画渲染过程

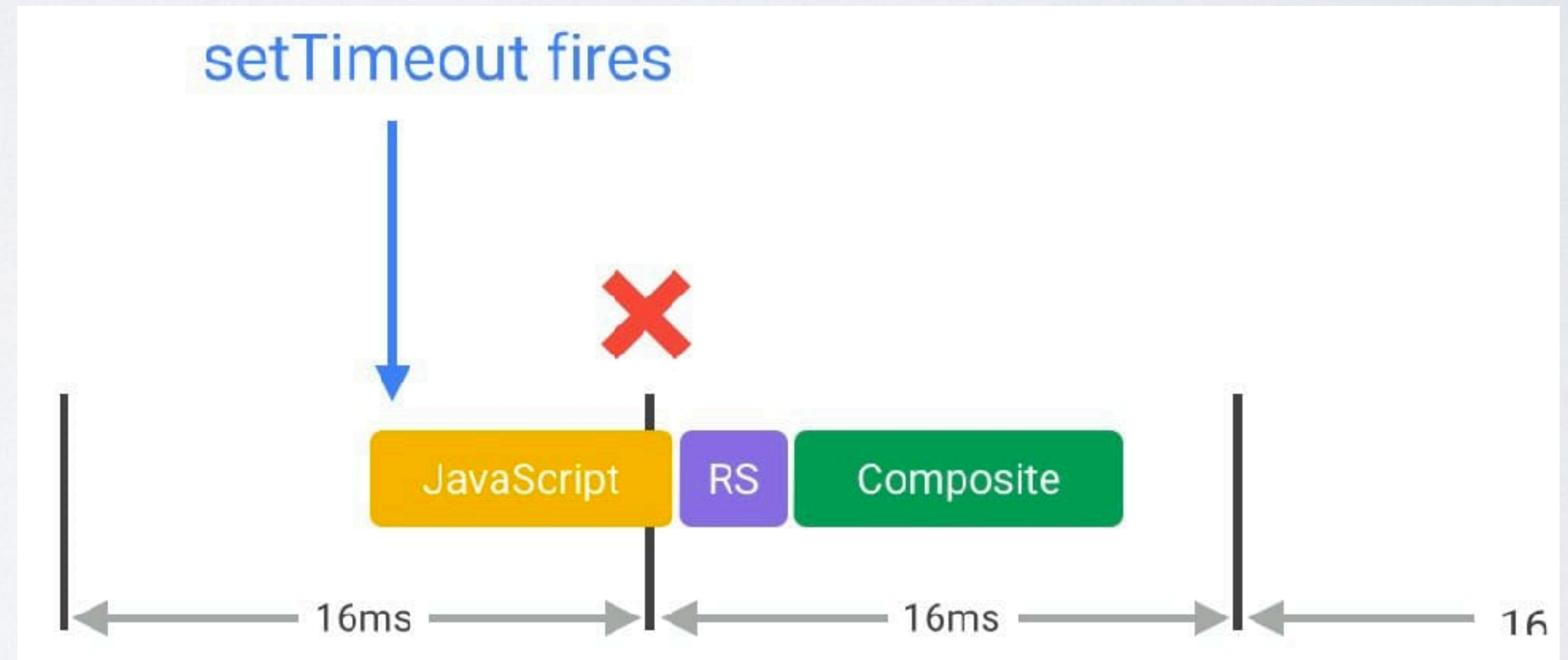


- 第二届 黄薇 [高性能CSS动画](#)
- 第一届 吴小倩 [谈谈CSS性能](#)
- [GoogleDeveloper](#) [渲染性能](#)



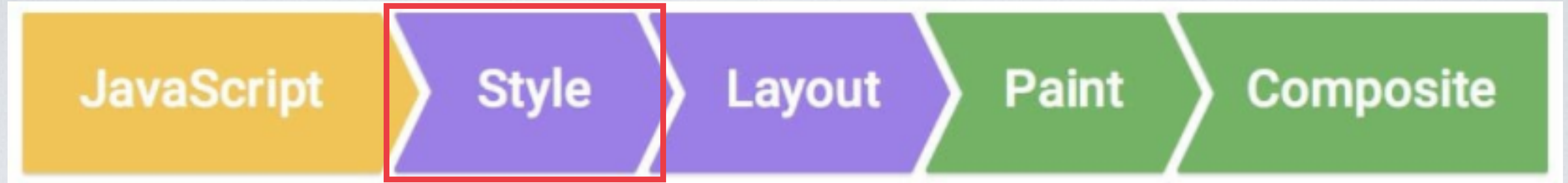
JS动画属性操作

- RequestAnimationFrame
- Web Worker
- 微任务
- 优化DOM读写操作
- DevTools



- [第二届 黄薇 高性能CSS动画](#)
- [第一届 吴小倩 谈谈CSS性能](#)
- [GoogleDeveloper 渲染性能](#)

动画渲染过程



样式重计算

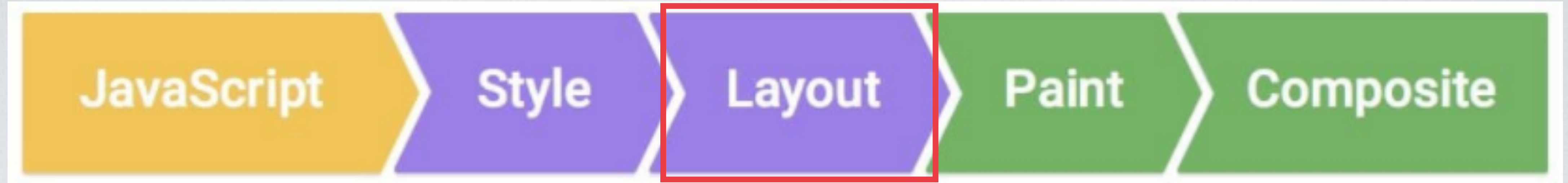
- 降低选择器复杂度 (BEM)
- 降低属性变更的元素数量

```
.box:nth-last-child(-n+1) .title {  
  /* styles */  
}
```



```
.final-box-title {  
  /* styles */  
}
```

- [第二届 黄薇 高性能CSS动画](#)
- [第一届 吴小倩 谈谈CSS性能](#)
- [GoogleDeveloper 渲染性能](#)

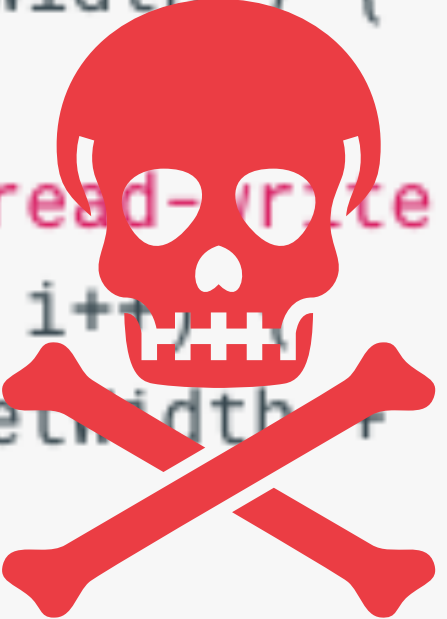


元素尺寸调整

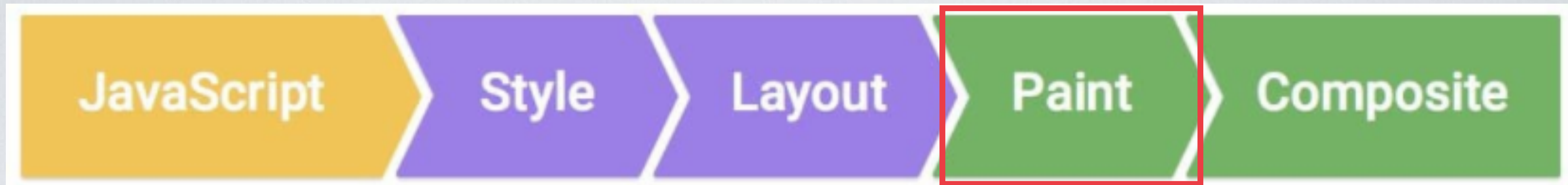
- 布局的作用范围是整个文档
- 使用Flexbox布局，而不是Float布局
- 读取样式也会触发Layout，缓存不变量，先读后写
- 使用RAF防抖处理

Reflow? 这**只在Firefox上**是这么称呼，需要改下了

```
function resizeAllParagraphsToMatchBlockWidth() {  
  
    // Puts the browser into a read-write-read-write cycle.  
    for (var i = 0; i < paragraphs.length; i++)  
        paragraphs[i].style.width = box.offsetWidth + 'px';  
}  
}
```

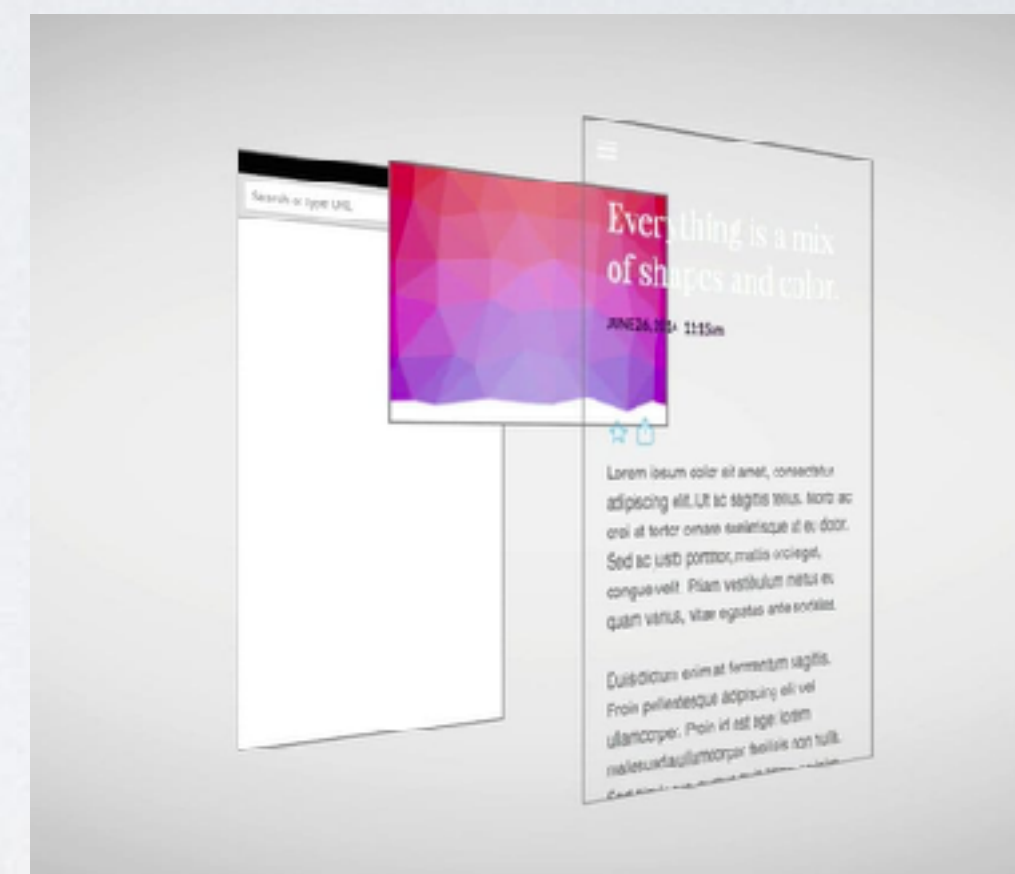


- [第二届 黄薇 高性能CSS动画](#)
- [第一届 吴小倩 谈谈CSS性能](#)
- [GoogleDeveloper 渲染性能](#)

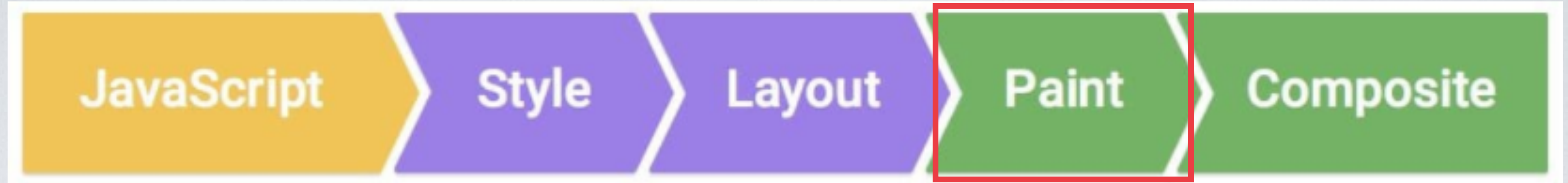


元素绘制

- 绘制是填充像素的过程，开销最大的部分
- transform 或 opacity 属性之外，更改任何属性始终都会触发绘制
- 创建层+动画编排减少绘制
- 使用绘制代价最小的元素
- DevTools找到瓶颈：Performance、Layout、Paint



- [第二届 黄薇 高性能CSS动画](#)
- [第一届 吴小倩 谈谈CSS性能](#)
- [GoogleDeveloper 渲染性能](#)



元素绘制-关于层的创建

- Chrome、Opera 和 Firefox -> **will-change**
- Safari 和 Mobile Safari -> **translateZ(0)**
- 层太多会造成内存管理开销
- **请勿在不分析的情况提升元素**

```
.moving-element {  
  will-change: transform;  
}
```

```
.moving-element {  
  transform: translateZ(0);  
}
```

- 第二届 黄薇 高性能CSS动画
- 第一届 吴小倩 谈谈CSS性能
- GoogleDeveloper 渲染性能

will-change属性

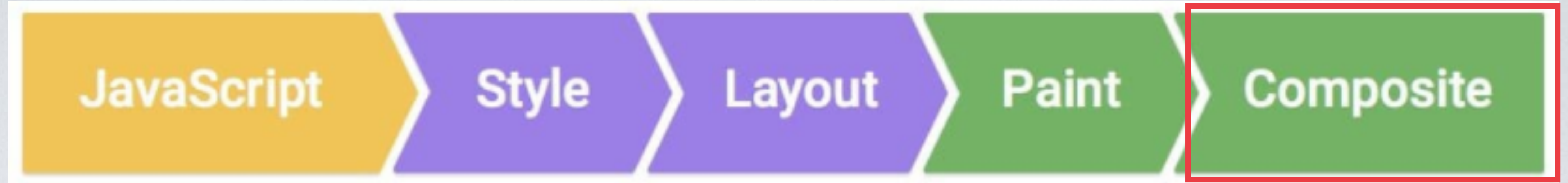
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android *	Chrome Android	UC for Android	QQ
			49								
			63								
		58	64			10.3		4.4			
11	16	59	65	11	50	11.2	all	62	64	11.8	1.2
	17	60	66	11.1	51	11.3					
		61	67	TP	52						
			68								



提前告知浏览器将要动画的元素

兼容性优点尴尬，但是写上又不会怀孕

- 第一届 吴小倩 [谈谈CSS性能](#)
- [CanIUse-willchange](#)



图层合并输出

- 动画部分创建层
- 使用transform 或 opacity 属性
- DevTools分析

合成是将页面的已绘制部分放在一起以在屏幕上显示的过程

- [第二届 黄薇 高性能CSS动画](#)
- [第一届 吴小倩 谈谈CSS性能](#)
- [GoogleDeveloper 渲染性能](#)

需要收藏的网址

Google Developers for Web

CSS渲染触发器

- 第二届 黄薇 高性能CSS动画
- 第一届 吴小倩 谈谈CSS性能
- GoogleDeveloper 渲染性能

4

动画

Web Animation API

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android	UC Browser for Android	QQ Browser
			² 49								
			² 63								
		³ 58	² 64			10.3		4.4			
11	16	59	² 65	⁴ 11	² 50	⁴ 11.2	all	² 62	² 64	² 11.8	² 1.2
	17	60	² 66	⁴ 11.1	² 51	⁴ 11.3					
		61	² 67	⁴ TP	² 52						
			² 68								

- 第一届 尤雨溪 CSS 与界面动效
- CanIUse Web Animation API

页脚有链接看效果

动画属性
Transform/
Animate/Matrix

动画属性
Canvas/WebGL









SVG动画

- 第三届 陈剑鑫 [从矩阵走入 WebGL 世界](#)
- 第二届 黄薇 [高性能CSS动画](#)
- 第一届 吴小倩 [谈谈CSS性能](#)
- 第一届 尤雨溪 [CSS 与界面动效](#)

- 第三届 陈剑鑫 [从矩阵走入 WebGL 世界](#)
- 第三届 方潇仪 [SVG动画实践](#)

5 技巧

不常用到的CSS3特性

 图像(Image)
 `<image>` `image()` `image-set()` `<gradient>` `linear-gradient()` `radial-gradient()` `repeating-linear-gradient()` `repeating-radial-gradient()`

- 第四届 王乐 [聊聊CSS中的黑科技](#)
- 第四届 城管 [CSS 黑魔法](#)
- 第三届 Wenting Zhang [CSS的隐藏绘画功能和交互动画技巧](#)
- 第二届 重拾 [CSS 的乐趣](#)
- developer.mozilla.org



6 工程化

工程化

主要是移动端工程化思路

工程化思路：为了满则**下面的规则**而进行的实践尝试并形成体系规范

工作流优化 -> 提高工作效率 -> **早点下班**

尝试的过程及结果：

屏幕等比缩放 -> rem-> 编码繁琐 ->

不同屏幕显示不同图片 -> 人肉切图上传改名字发布 ->

规范化开发约束自由发挥 -> 项目初始化工具

px2rem

lib.img

start-kit

autoprefixer

img-upload

6

工程化

游击队 v.s. 正规军

是不是很简单

做项目有积累

能**主动思考**目前工作中有哪些是能优化的地方，然后改进

7 新规范

7

新规范

Houdini怎么玩呢?

1. 下载: Chrome Canary
2. 打开特性表: <chrome://flags/#enable-experimental-web-platform-features>
3. 重启

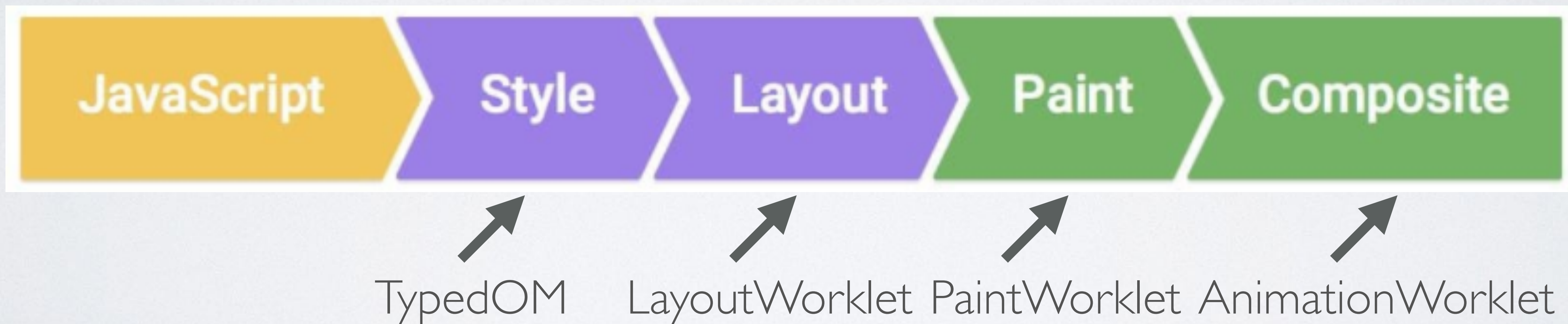
7 新规范

Houdini新规范解决的问题及所处的位置

TypedOM: 属性能定义类型, 能获取正确的属性值

CSS Properties and Values API: 为CSS Variable做类型增强

Worklets: 创造主线程之外的渲染线程



7 新规范

DEMO

7 新规范

CSS Grid Layout

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android *	Chrome Android	UC for Android	QQ
			1 49								
			63								
		58	64			10.3		4.4			
2 11	16	59	65	11	50	11.2	all	62	64	11.8	1 1.2
	17	60	66	11.1	51	11.3					
		61	67	TP	52						
			68								

CSS Grid Layout IE10+, 建议从Edge16(2017/10)开始
Android 62(2017/01), IOS 10.3(2017/03)

希望我们前端对**B端**项目的浏览器兼容方面能硬气点

- 第三届 大漠 [CSS Grid Layout](#)
- [CanIUse-CSS Grid Layout](#)

新规范

FlexBox, GridLayout?

FlexBox(IE10+): 定义一个维度, 行 or 列, 替换Float/Table布局
GridLayout: 定义两个维度, 行 and 列, 替换目前的栅格系统

- 第三届 大漠 [CSS Grid Layout](#)
- [CanIUse-CSS Grid Layout](#)

7

新规范

CSS Grid Layout

一堆属性，带大家过一遍API浪费大家时间

照着示例练一遍

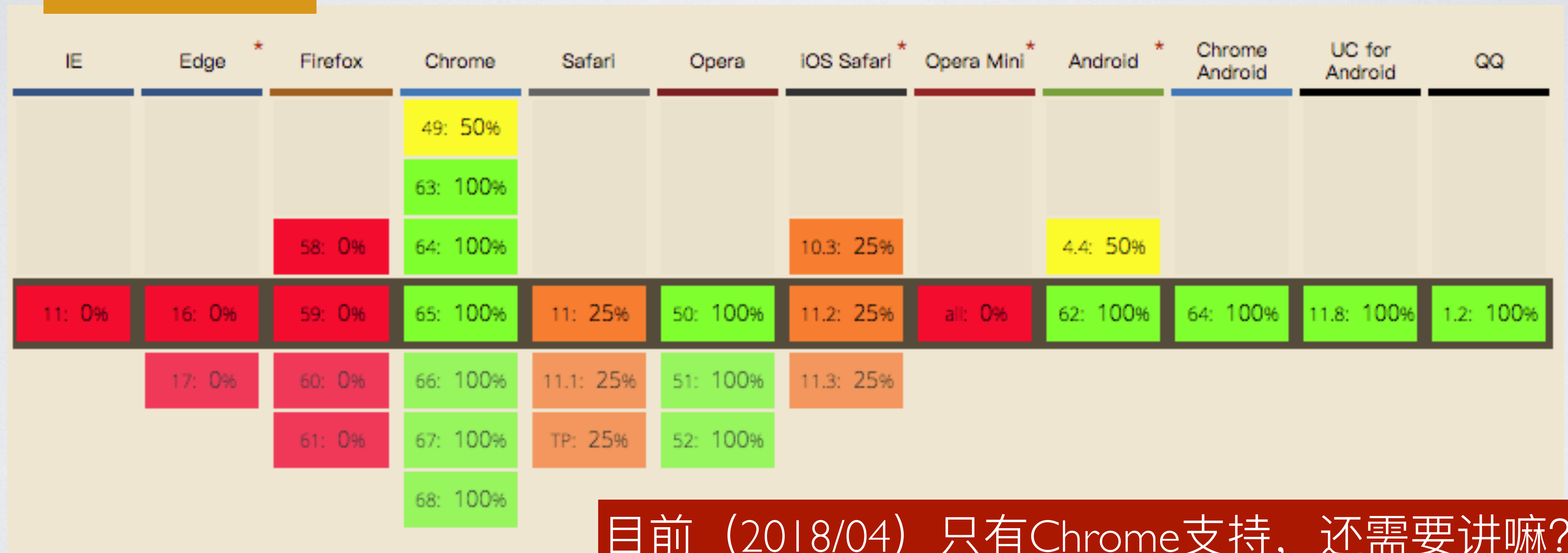
Grid By Example

- 第三届 大漠 CSS Grid Layout
- CanIUse-CSS Grid Layout
- gridbyexample.com

7 新规范

Web-Component

Shadow DOM



目前 (2018/04) 只有Chrome支持, 还需要讲嘛?

- 第一届 勾三股四 Web Components中的CSS
- CanIUse-ShadowDOM

8 其他

技巧太多，推荐几本书



张鑫旭 《CSS世界》



CSS魔法 《CSS揭秘》

强烈推荐

- 第二届 张鑫旭 Leader, 我不想写CSS
- 第四届 CSS魔法 重拾 CSS 的乐趣

进阶技巧

认真读下两本书

每个点写下DEMO练手，关键是抓住本质（联想拓展）

深刻理解CSS在浏览器渲染流程中的位置

关注新规范

熟练使用开发者工具调试CSS性能

做到心中有张地图

CSS-IN-JS 《A Unified Styling Language》

I. Scoped Styles

传统:

Global -> BEM/OOCSS/SMACSS命名约定

CSS-IN-JS:

CSSModule -> Glamor -> JSS ->

Styled-Components解析字符串 -> Glamorous对象解析

开发效率低，脱离结构和表现分离的模型
以上CSS-IN-JS都是针对React，Vue没这个问题
动态样式可以使用新规范解决(IE不支持)

CSS-IN-JS 《A Unified Styling Language》

2. Critical CSS

首屏**CSS**资源内嵌加快响应速度，可用在SSR场景

这个应该是通过工具解决的问题
人肉通过编码解决不优雅

CSS-IN-JS 《A Unified Styling Language》

3. Smarter Optimisations

对**CSS**智能优化，比如生成**Atomic CSS**，做到体积最小

开发模式需要切到普通模式，否则调试痛苦
线上问题排除可能是痛点
不利于浏览器插件开发
动画性能是个问题

CSS-IN-JS 《A Unified Styling Language》

4. NPM Package Manager

将**CSS**做成**JS**模式

Webpack也能支持引入CSS资源，加上Loader就行

CSS-IN-JS 《A Unified Styling Language》

5. Non-Browser Styling

在React框架下，可用过React-DOM转到浏览器平台，
通过React-Native转到App平台

保证兼容会舍弃一些特性
创新点，赞成！

CSS-IN-JS 《A Unified Styling Language》

Conclusion

需要时再学，
CSS-IN-JS可充分解决CSS在React中的问题，额
但不是完全必要。
保持关注

充分但不必要

我们是软件工程师

共性问题通过工具解决，别“Don't Repeat Yourself”

精力够的话再朝工程化方向发展，解决大多数人的效率问题

8

其他

给自己一个月时间成为CSS大牛

前端很简单，方法很重要
按照分类自己尝试找资料总结+实践

听者反馈

“看完他的报告就别再去CSS大会了，浪费钱！”

—特朗普

“恭喜你，在短短1小时搞定21个CSS大会报告！”

—罗振宇 得到APP



Q&A